Machine Learning 1

Lecture 13.3 - Combining Models
Boosting

Erik Bekkers

(Bishop 14.3)

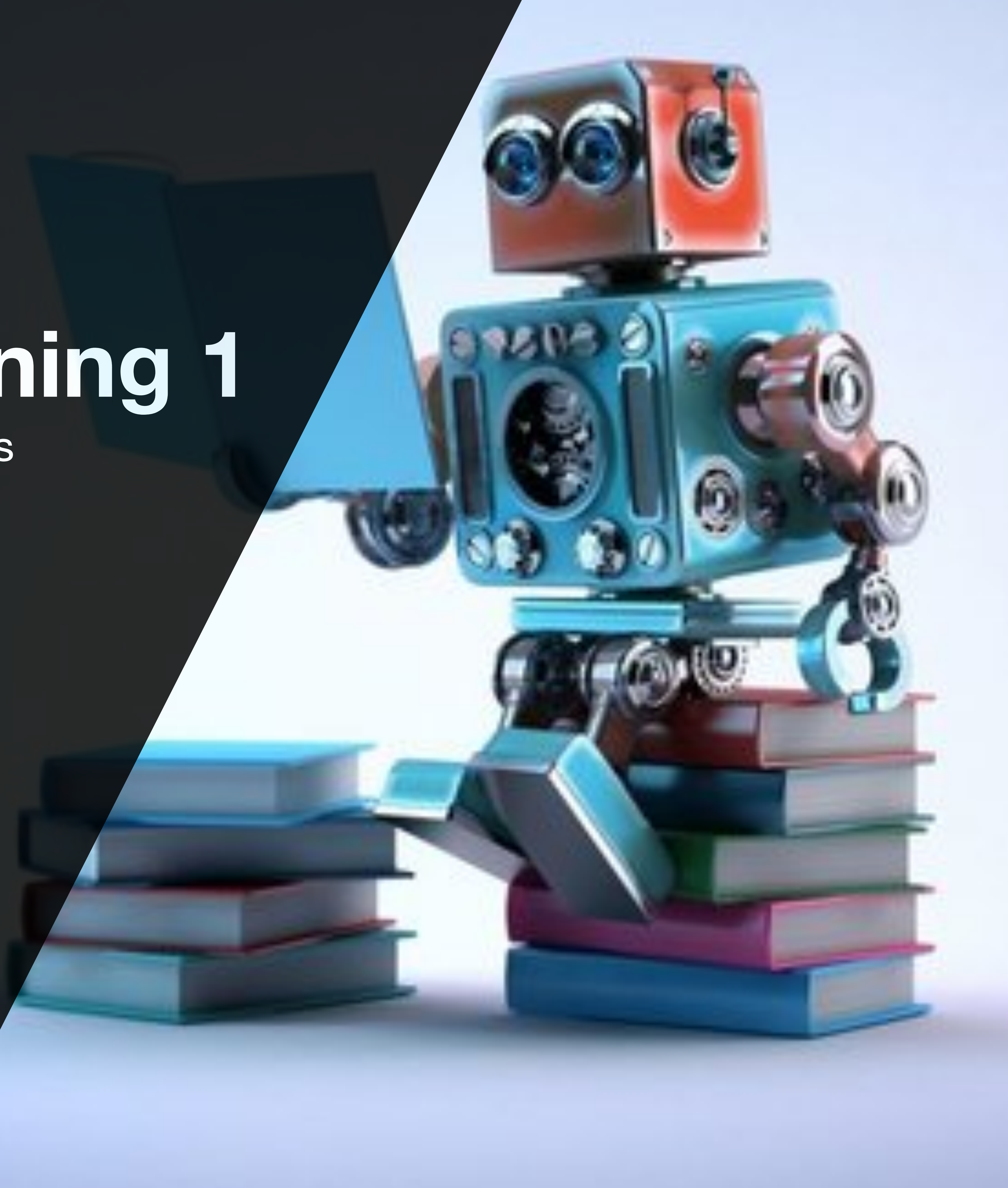


Image credit: Kirillm | Getty Images

Slide credits: Patrick Forré and Rianne van den Berg

Regression with GP’s

- Combining models: (Bishop 4.1-4.4)
  - Bayesian model averaging vs. model combination methods
  - **Committees**:
    - Bootstrap aggregation
    - Random subspace methods
    - **Boosting**
  - Decision trees
  - Random forests

Boosting

- Committee consists of multiple base classifiers
- The performance of the committee can be significantly better than that of any of the base classifiers
- **AdaBoost: adaptive boosting**
- Boosting can give good results even if the base classifiers have a performance that is only slightly better than random
- Base classifiers are simple models/*weak learners*
- Can also be extended to regression.

bootstrapping / bagging : decreasing variance

boosting : decreasing bias (and variance)

Boosting

- Base classifiers are trained in sequence
- Note this contrast with other committee methods such as bagging
- Each base classifier is trained using a weighted form of the dataset
- The weighting coefficient associated with each datapoint depends on the performance of previous classifiers. loss: $(\underline{x}_n, t_n, w_n)$
- Points that are misclassified by one of the base classifiers are given greater weight when used to train the next base classifier in the sequence.
- When all classifiers are trained, predictions are combined through a weighted majority voting scheme.

$\in \{-1, 1\}$

$$Y_M(\mathbf{x}) = \text{sign}\left(\sum_{m=1}^{M} \alpha_m y_m(\mathbf{x})\right)$$

weight on model

high α ← good performing model

Boosting: binary classification

- Dataset $\{(\mathbf{x}_n, t_n)\}_{n=1}^{N}$ with $t_n \in \{-1, +1\}$
- Each data point has an associated weighting parameter w_n
- The weights are initialized to $w_n = 1/N$
- We assume we have a procedure to train a base classifier m such that it produces a function $y_m(\mathbf{x}) \in \{-1, +1\}$
- Adaboost:
  - At each stage a new classifier is trained on weighted dataset
  - Weights for data points that were misclassified by previous classifier are increased
  - When all classifiers are trained, committee is formed by weighted base classifiers

Adaboost

1. Initialize weights: $w_n^{(1)} = 1/N$ for $n = 1,\ldots,N$
2. for $m = 1,\ldots,M$:

   indicator function

   (a) Fit classifier $y_m(\mathbf{x})$ to minimize $J_m = \sum_{n=1}^{N} w_n^{(m)} I[y_m(\mathbf{x}_n) \neq t_n]$

   (b) compute weighted error rates $\epsilon_m = \dfrac{\sum_{n=1}^{N} w_n^{(m)} I[y_m(\mathbf{x}_n) \neq t_n]}{\sum_{n=1}^{N} w_n^{(m)}}$

   model weight

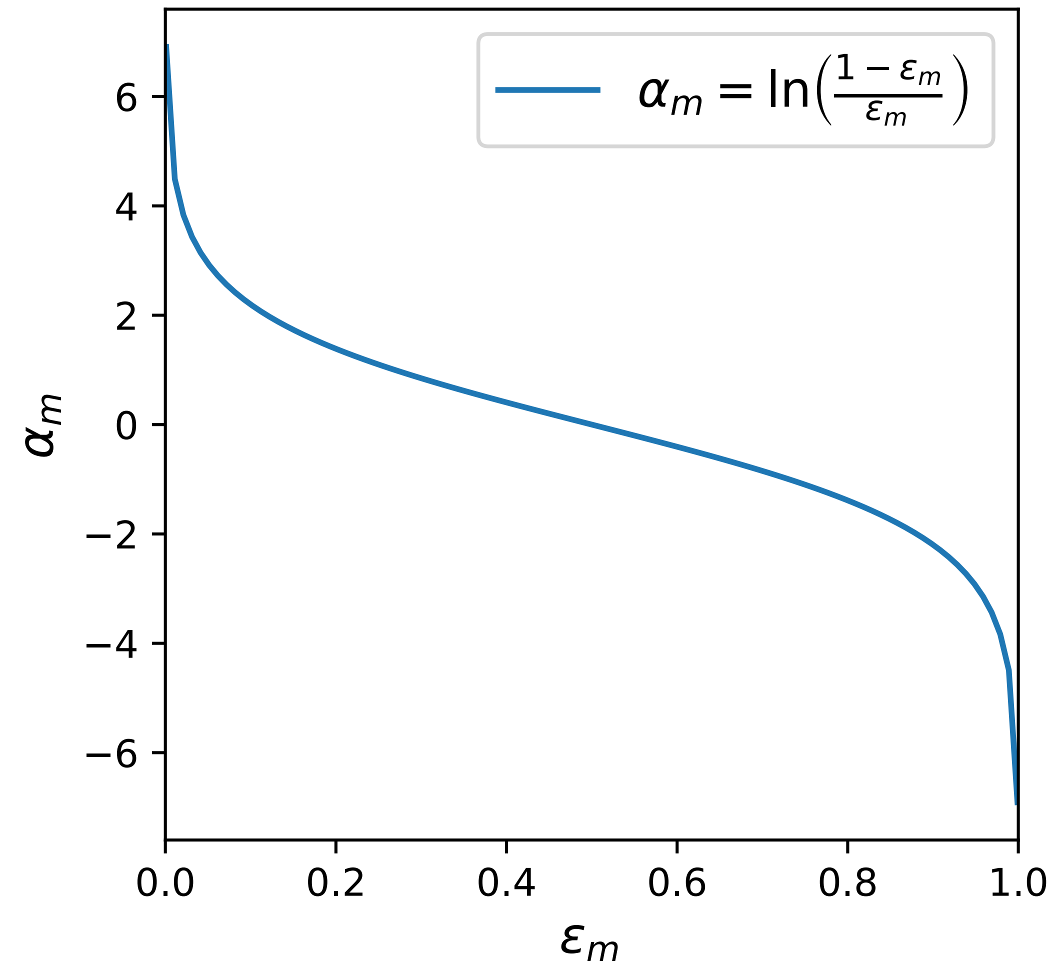
   and $\alpha_m = \ln\left(\dfrac{1-\epsilon_m}{\epsilon_m}\right)$

   (c) Update weights $w_n^{(m+1)} = w_n^{(m)} \exp\{\alpha_m I[y_m(\mathbf{x}_n) \neq t_n]\}$

3. Make predictions $Y_M(\mathbf{x}) = \text{sign}\left(\sum_{m=1}^{M} \alpha_m y_m(\mathbf{x})\right)$

Adaboost

- Prediction $Y_M(\mathbf{x}) = \operatorname{sign}\left(\sum_{m=1}^{M} \alpha_m y_m(\mathbf{x})\right)$
- prediction weights $\alpha_m = \ln\left(\frac{1-\epsilon_m}{\epsilon_m}\right)$
- weighted error rates $\epsilon_m = \frac{\sum_{n=1}^{N} w_n^{(m)} I[y_m(\mathbf{x}_n) \neq t_n]}{\sum_{n=1}^{N} w_n^{(m)}}$
- Greater weights for more accurate classifiers!

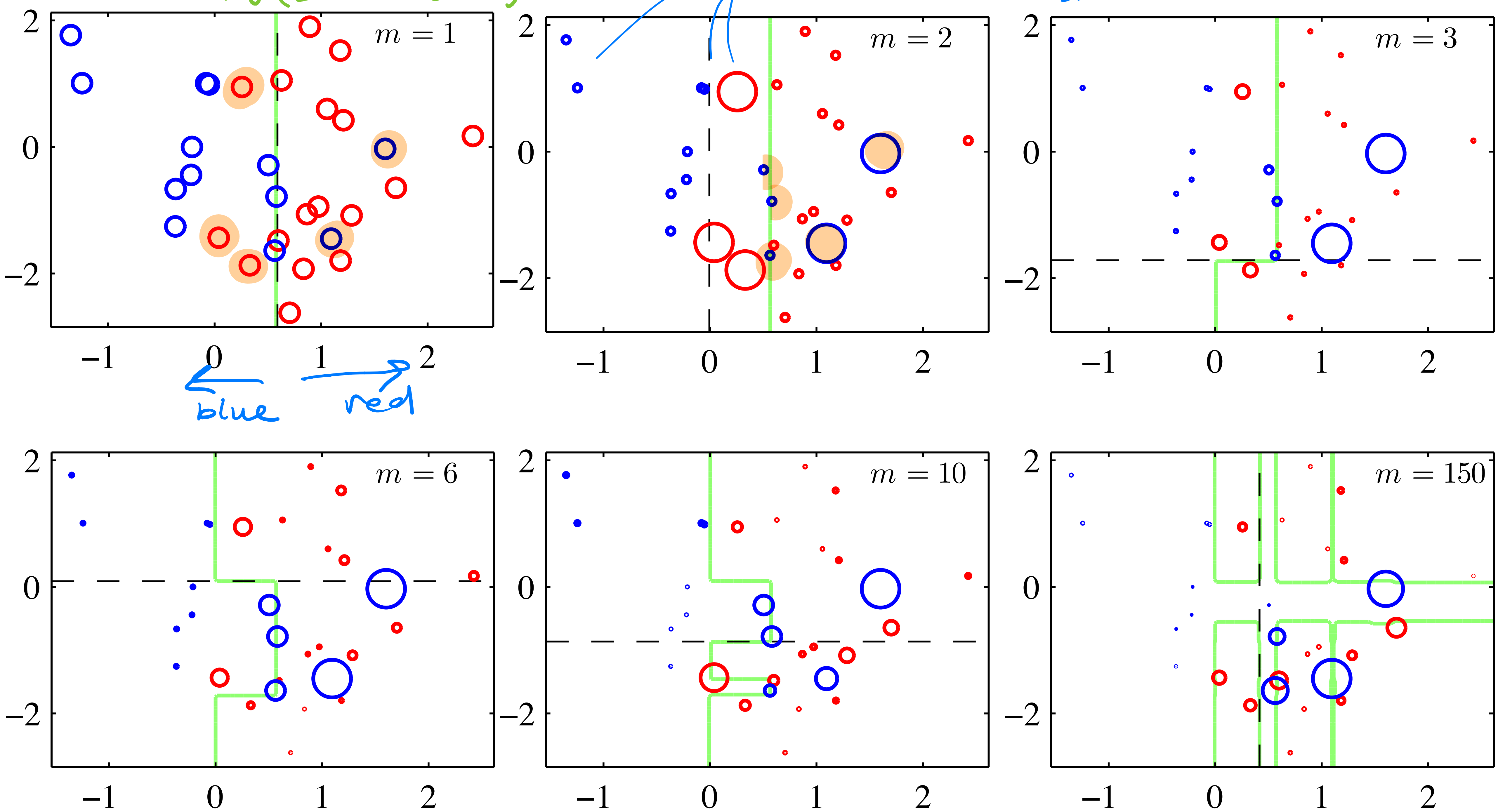


Boosting example: decision stumps

--- y_m

⟶ $\text{sign}\left(\sum \alpha_m y_m\right)$

$\tilde{w}_n^{(m)} = w_n^{(m)} / \sum_{n=1}^{N} w_n^{(m)}$



Interpretation of Adaboost

- Sequential minimization of exponential error function
- Error function $E_m = \sum_{n=1}^{N} \exp\{-\overbrace{t_n f_m(\mathbf{x}_n)}^{>0 \text{ if correct}}\}$

  ← data points

- Linear combination of base classifiers $y_l(\mathbf{x})$

  ← committee members

$$f_m(\mathbf{x}) = \frac{1}{2} \sum_{l=1}^{m} \alpha_l y_l(\mathbf{x})$$

- Goal: minimize E with respect to $\{\alpha_l\}$ and parameters of base classifiers $y_l(\mathbf{x})$
- Sequential minimization:
  - Fix parameters of $y_1(\mathbf{x}), \ldots, y_{m-1}(\mathbf{x})$ and $\alpha_1, \ldots, \alpha_{m-1}$
  - Minimize E w.r.t. parameters of $y_m(\mathbf{x})$ and α_m

Derivation of Adaboost

$f_m = \frac{1}{2}\sum_{l=1}^{m} \alpha_l y_l(\underline{x}_n)$

- Error function

$$E_m = \sum_{n=1}^{N} \exp\{-t_n f_m(\mathbf{x}_n)\} = \sum_{n=1}^{N} \exp\{-t_n f_{m-1}(\mathbf{x}_n) - \frac{1}{2} t_n \alpha_m y_m(\mathbf{x}_n)\}$$

$$= \sum_{n=1}^{N} w_n^{(m)} \exp\{-\frac{1}{2} t_n \alpha_m y_m(\mathbf{x}_n)\}, \qquad w_n^{(m)} = \exp(-t_n f_{m-1}(x_n))$$

- Let T_m be the set of correctly classified data points $y_m(\mathbf{x})$ $\quad (t_n y_m(\mathbf{x}_n) = 1)$
- Let M_m be the set of mis- ~~correctly~~ classified data points $y_m(\mathbf{x})$ $\quad (t_n y_m(\mathbf{x}_n) = -1)$
- Error function:

$$E_m = e^{-\alpha_m/2} \sum_{n \in T_m} w_n^{(m)} + e^{+\alpha_m/2} \sum_{n \in M_m} w_n^{(m)}$$

$$= \left(e^{\alpha_m/2} - e^{-\alpha_m/2}\right) \sum_{n=1}^{N} w_n^{(m)} I[y_m(\mathbf{x}_n) \neq t_n] + e^{-\alpha_m/2} \sum_{n=1}^{N} w_n^{(m)}$$

Derivation of Adaboost

- Error function

$$E_m = \left(e^{\alpha_m/2} - e^{-\alpha_m/2}\right) \sum_{n=1}^{N} w_n^{(m)} I[y_m(\mathbf{x}_n) \neq t_n] + e^{-\alpha_m/2} \sum_{n=1}^{N} w_n^{(m)}$$

- Minimization w.r.t. $y_m(\mathbf{x})$ minimizes

(a)

$$J_m = \sum_{n=1}^{N} w_n^{(m)} I[y_m(\mathbf{x}_n) \neq t_n]$$

- Minimization w.r.t. α_m: $\dfrac{\partial E_m}{\partial \alpha_m} = 0$

(b)

$$\alpha_m = \ln\left(\frac{1-\epsilon_m}{\epsilon_m}\right) \qquad \epsilon_m = \frac{\sum_{n=1}^{N} w_n^{(m)} I[y_m(\mathbf{x}_n) \neq t_n]}{\sum_{n=1}^{N} w_n^{(m)}}$$

Derivation of Adaboost

- After we found $y_m(\mathbf{x})$ and α_m, we minimize E_{m+1} w.r.t. $y_{m+1}(\mathbf{x})$ and α_{m+1}

$$E_{m+1} = \sum_{n=1}^{N} \exp\{-t_n f_{m+1}(\mathbf{x}_n)\}$$

$$= \sum_{n=1}^{N} \exp\{\overbrace{\underline{-t_n f_{m-1}(\mathbf{x}_n)} - \frac{1}{2} t_n \alpha_m y_m(\mathbf{x}_n)}^{-t_n f_m} - \frac{1}{2} t_n \alpha_{m+1} y_{m+1}(\mathbf{x}_n)\}$$

$$= \sum_{n=1}^{N} \underline{w_n^{(m)}} \exp\{-\frac{1}{2} t_n \alpha_m y_m(\mathbf{x}_n)\} \exp\{-\frac{1}{2} t_n \alpha_{m+1} y_{m+1}(\mathbf{x}_n)\}$$

$$= \sum_{n=1}^{N} w_n^{(m+1)} \exp\{-\frac{1}{2} t_n \alpha_{m+1} y_{m+1}(\mathbf{x}_n)\}$$

(c)
- Weights are updated: $w_n^{(m+1)} = w_n^{(m)} \exp\{-\frac{1}{2} t_n \alpha_m y_m(\mathbf{x}_n)\}$

Derivation of Adaboost

- Weight updates $w_n^{(m+1)} = w_n^{(m)} \exp\{-\frac{1}{2} t_n \alpha_m y_m(\mathbf{x}_n)\}$
- Use $t_n y_m(\mathbf{x}_n) = 1 - 2I[y_m(\mathbf{x}_n) \neq t_n]$
- Then $w_n^{(m+1)} = w_n^{(m)} \exp\{\cancel{-\alpha_m}/2\}\exp\{\alpha_m I[y_m(\mathbf{x}_n) \neq t_n]\}$ (c)

  does not depend on n
- Term $\exp\{-\alpha_m/2\}$ is independent of n and can be discarded
- Prediction:

$$\mathrm{sign}\left(f_m(\mathbf{x})\right) = \mathrm{sign}\left(\frac{1}{2}\sum_{l=1}^{m} \alpha_l y_l(\mathbf{x})\right) = \boxed{\mathrm{sign}\left(\sum_{l=1}^{m} \alpha_l y_l(\mathbf{x})\right)}$$

3.

Advantages and disadvantages

- Exponential error function makes Adaboost very simple algorithm
- Very sensitive to outliers for which $t_n y_m(\mathbf{x})$ is large negative
- Exponential error function cannot be interpreted as log-likelihood of well-defined probabilistic model
- Doesn't generalize straightforwardly to $K > 2$